MAX-PLANCK-GESELLSCHAFT

Peter Benner Patrick Kürschner Jens Saak

Efficient Handling of Complex Shift Parameters in the Low-Rank Cholesky Factor ADI method

MAX–PLANCK–INSTITUT
FÜR DYNAMIK KOMPLEXER
TECHNISCHER SYSTEME
MAGDEBURG

Max Planck Institute Magdeburg
Preprints

MPIMD/11-08 March 16, 2012

Impressum:

Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg

Publisher:
Max Planck Institute for Dynamics of Complex Technical Systems

Address:
Max Planck Institute for Dynamics of Complex Technical Systems
Sandtorstr. 1
39106 Magdeburg

www.mpi-magdeburg.mpg.de/preprints

Efficient Handling of Complex Shift Parameters in the Low-Rank Cholesky Factor ADI method.

Peter Benner, Patrick Kürschner, and Jens Saak

Abstract

The solution of large-scale Lyapunov equations is a crucial problem for several fields of modern applied mathematics. The low-rank Cholesky factor version of the alternating directions implicit method (LRCF-ADI) is one iterative algorithm that computes approximate low-rank factors of the solution. In order to achieve fast convergence it requires adequate shift parameters, which can be complex if the matrices defining the Lyapunov equation are unsymmetric. This will require complex arithmetic computations as well as storage of complex data and thus, increase the overall complexity and memory requirements of the method. In this article we propose a novel reformulation of LRCF-ADI which generates real low-rank factors by carefully exploiting the dependencies of the iterates with respect to pairs of complex conjugate shift parameters. It significantly reduces the amount of complex arithmetic calculations and requirements for complex storage. It is hence often superior in terms of efficiency compared to other real formulations.

I. Introduction

Lyapunov matrix equations arise in numerous fields related to control theory. They are an important tool, for instance, for stability analysis and stabilization [17], model order reduction [25] and in Newton type methods for solving algebraic Riccati equations [21]. Here we investigate the Lyapunov equation

$$AP + PA^T = -BB^T \tag{1}$$

with $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. If A is Hurwitz, i.e., the spectrum $\Lambda(A)$ lies in the open left complex half plane, which is in the remainder denoted by $\mathbb{C}_-$, there exists a unique symmetric positive semidefinite solution $P \in \mathbb{R}^{n\times n}$. We assume that the number of columns m of B is much smaller then the dimension n of A. For small to moderately sized A the Lyapunov equation can be solved by direct methods such as, e.g., the Bartels-Stewart algorithm [2], Hammerling's method [20], the so called $2-$solve schemes [33] of both, and the matrix sign function iteration [28]. Since we are interested in the case where A is large and sparse, we investigate an iterative algorithm based on the alternating directions implicit (ADI) method which is capable of solving large-scale Lyapunov equations. There, low-rank factors $\tilde{Z} \in \mathbb{C}^{n\times r}$ with $r \ll n$ are computed such that $\tilde{Z}\tilde{Z}^H \approx P$. This is motivated by the observation that the singular values of P decay quite rapidly in many applications [32], [19]. The ADI method, and its low-rank version, need a number of shift parameters to achieve fast convergence. It can be shown that globally optimal shift parameters are the solutions of a certain rational minimax problem and there are several approaches to compute, or at least approximate, these optimal values. For unsymmetric A, the case we draw special emphasis to, it is likely that complex shift parameters need to be applied, which will require complex arithmetics and produce complex low-rank factors. This is our main concern in this paper and we propose a modification of the low-rank ADI that allows the generation of real factors even if complex shift parameters have to be used. Furthermore, although the method still employs complex core computations, their amount is reduced significantly, as well as the required amount of memory due to complex data types.

The remainder of this paper is organized as follows. In Section II we review the low-rank Cholesky factor ADI method as well as the required shift parameters. Section III shows previous approaches that deal with complex shift parameters and introduces our new approach to this problem. This modification can also be implemented in the ADI method for generalized Lyapunov equations and in the Newton type algorithms for algebraic Riccati equations, as it is described in Section IV. The efficiency of the proposed approach for (generalized) Lyapunov and algebraic

Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstraße 1, 39106 Magdeburg, Germany, {benner, kuerschner, saak}@mpi-magdeburg.mpg.de, Phone: +49 391 6110 {450, 424}, Fax: +49 391 6110 453, correspondence and proof return to P. Kürschner

Algorithm 1 Low-rank Cholesky factor ADI iteration (LRCF-ADI)

```
Input: $A$ and $B$ as in (1) and shift parameters $\{\mu_1, \dots, \mu_{j_{max}}\}$.
Output: $Z = Z_{jmax} \in \mathbb{C}^{n \times t_{j_{\max}}}$, such that $ZZ^H \approx P$
 1: for $j = 1, 2, \dots, j_{max}$ do
 2:   if $j = 1$ then
 3:     Solve $(A + \mu_1 I_n)V_1 = \sqrt{-2\,\mathrm{Re}\,(\mu_1)}B$ for $V_1$.
 4:     $Z_1 = V_1$.
 5:   else
 6:     Solve $(A + \mu_j I_n)\tilde{V} = V_{j-1}$ for $\tilde{V}$.
 7:     $V_j = \sqrt{\mathrm{Re}\,(\mu_j)/\,\mathrm{Re}\,(\mu_{j-1})}\left(V_{j-1} - (\mu_j + \overline{\mu_{j-1}})\tilde{V}\right)$.
 8:     Update LRCF $Z_j = [Z_{j-1}, \quad V_j]$.
 9:   end if
10: end for
```

Riccati equations with large and sparse matrices is illustrated in Section V. Section VI concludes and gives some possible further related research perspectives.

We use the following notation in this paper: $\mathbb{R}$ and $\mathbb{C}$ denote the real and complex numbers, and $\mathbb{R}_-$, $\mathbb{C}_-$ refer to the set of strictly negative numbers and the open left half plane. In the matrix case, $\mathbb{R}^{n\times m}$, $\mathbb{C}^{n\times m}$ denote $n \times m$ real and complex matrices, respectively. For any complex quantity $X = \mathrm{Re}\,(X) + \jmath\,\mathrm{Im}\,(X)$, $\mathrm{Re}\,(X)$, $\mathrm{Im}\,(X)$ are its real and imaginary parts, and $\jmath$ is the imaginary unit. The complex conjugate of X is denoted by $\overline{X} = \mathrm{Re}\,(X) - \jmath\,\mathrm{Im}\,(X)$. The absolute value of $\xi \in \mathbb{C}$ is denoted by $|\xi|$. The matrix A^T is the transpose of a real $n \times m$ matrix, and $A^H = \overline{A}^T$ is the complex conjugate transpose of a complex matrix. The identity matrix of dimension n is indicated by I_n.

II. The low-rank Cholesky factor ADI method

For (1) the ADI iteration [37] is given by

$$
\begin{aligned}
(A + \mu_j I_n)P_{j-\frac{1}{2}} &= -BB^T - P_{j-1}(A^T - \mu_j I_n) \\
(A + \mu_j I_n)P_j^T &= -BB^T - P_{j-\frac{1}{2}}^T(A^T - \mu_j I_n),
\end{aligned}
$$

where P_j, $j \in \mathbb{N}_0$ denote the j-th approximations of P and $\mu_j \in \mathbb{C}_-$ are suitable shift parameters. Although working with an initial guess P_0 is possible, we use only $P_0 = 0$ in the sequel. Rewriting the double step into one single step and setting $P_j = Z_j Z_j^T$ leads after some basic manipulations (see, e.g., [30]) to the low-rank Cholesky factor ADI method [26], [22], as given in Algorithm 1. We see that in each iteration m new columns are added to the approximate low-rank factor, such that after J iterations we have Jm columns in Z_J. As stated in the introduction, we assume that m is much smaller than n. The solution of the linear systems with the shifted matrices $A + \mu_j I_n$ is the most expensive step in the (LRCF)-ADI, and we assume that we are able to solve these systems by sparse direct [14], [13] or iterative solvers [29], [36]. If m is large, such that the linear systems have many right hand sides, the use of iterative solvers is severely restricted.

A. Shift Parameters

For J iterations of (LRCF)-ADI, the optimal set of shift parameter can be related to a rational minimax problem [38] which involves the complete set of eigenvalues of A. For large matrices the exact eigenvalues are not so easily available, and therefore one uses a small number of approximate eigenvalues in the minimax problem. An often used inexpensive approach is carried out by taking $k_+ \ll n$ Ritz values of A and, additionally, the reciprocals of $k_- \ll n$ Ritz values of A^{-1}. Both sets of Ritz values are obtained using an Arnoldi process. The shift parameters generated using this approach are often referred to as heuristic shifts, or sometimes Penzl shifts in honor of the author of the article [26], where their first appearance can be found.

In the context of model order reduction, some numerical results (e.g., in [30]) suggest to include some *dominant poles* [23] in the set of shift parameters to achieve higher accuracies in the reduced order models. Dominant poles refer to the eigenvalues λ of A with corresponding right and left eigenvectors $0 \neq x,\ y \in \mathbb{C}^n$, where the quantity $\hat{R} := \| \frac{(C^T x)(y^T B)}{\mathrm{Re}\,(\lambda)} \|_2$ is large compared to the other eigenvalues and -vectors. The matrix $C \in \mathbb{R}^{p \times n}$ denotes the output mapping of the processed dynamical system.

If A is unsymmetric some of its eigenvalues might be complex and come in conjugate pairs, and therefore it is possible that the same happens for some of the shift parameters. This will of course lead to complex low-rank factors, and since one complex multiplication involves three real multiplications the overall computational effort is increased.

For some problems, for example balanced truncation model order reduction [25], one is intrinsically interested in real LRCFs, and from a numerical point of view these should preferably be obtained by an algorithm whose computations and storage requirements stay completely in real arithmetics. In the next part of this work we describe different approaches to achieve this goal completely or at least partially. We propose a new approach which generates real LRCFs and decreases the computational effort introduced by complex shift parameters significantly by reducing the number of complex linear systems which need to be processed. All approaches require the natural and important convention that the set of shift parameters is *proper*, i.e., it is closed with respect to complex conjugation and of the form

$$\{\mu_1, \ldots, \mu_J\} = \{\nu_1, \ldots, \nu_L\} \subset \mathbb{C}_-,$$

where ν_j is either a negative real number or a pair of complex conjugate numbers with negative real parts. This structure automatically implies that $ZZ^H \in \mathbb{R}^{n \times Jm}$. Throughout the rest of this paper we also stick to proper parameter sets. If the total number of iterations exceeds the number $j_{\max}$ of available shift parameters, then the shift parameters are usually used in a cyclic manner, i.e., $\mu_j = \mu_{(j \mod j_{\max})+1},\ j \in \mathbb{N}$.

B. Stopping Criteria

There are several ways to terminate Algorithm 1, for example when a certain number $j_{\max}$ of iterations is reached or the current approximation $ZZ^H \approx P$ is of a sufficient accuracy. In the latter case one can, e.g., check if the normalized residual norm

$$\|AZZ^H + ZZ^H A^T + BB^T\|_2 / \|BB^T\|_2 \tag{2}$$

of the approximation is smaller than a prescribed tolerance $0 < \epsilon_{\text{res}} \ll 1$. Since the residual is a symmetric matrix, it is suggested to use its largest eigenvalue, which is also its $\|\cdot\|_2$-norm. This can be computed via a Lanczos-, or Arnoldi method. However, if the norm of BB^T is too small, it might be reasonable to choose another normalization in (2), e.g., dividing by $\|BB^T\|_2 + 2\|A\|_2 \|ZZ^H\|_2$.

Another stopping criterion is based on the relative change in the LRCF Z_j. For a small constant $0 < \epsilon_{\text{rc}} \ll 1$, the iteration stops if

$$\|V_j\|_F / \|Z_j\|_F \leq \epsilon_{\text{rc}}, \tag{3}$$

where it is not necessary to compute $\|Z_j\|_F$ every time, because $\|Z_j\|_F^2 = \|Z_{j-1}\|_F^2 + \|V_j\|_F^2$ allows to accumulate it, leaving the computation of the Frobenius norm of V_j only.

C. Further Improvements

Although not further discussed in this paper, we mention a few strategies to increase the efficiency of LRCF-ADI both with respect to convergence speed as well as memory usage.

1) Acceleration via Galerkin projection: Krylov subspace methods represent another class of iterative algorithms for solving for large-scale Lyapunov equations (see, e.g., [31] and the references therein). If A is dissipative, i.e., $A + A^T$ is negative definite, those methods perform a Galerkin style projection to (1). In order to accelerate the convergence of LRCF-ADI, this motivates to carry out a similar projection [30], [10] onto the column space of the current LRCF approximation Z. For reasons of numerical stability an orthonormal basis for $\mathrm{range}\,(Z)$ is used which requires an orthonormalization of Z. To reduce the induced extra amount of work this projection is usually only invoked after a couple of iterations. Let S contain the basis vectors as columns of this orthonormal base. The projected Lyapunov equation is then

$$\hat{A}\hat{P} + \hat{P}\hat{A}^T = -\hat{B}\hat{B}^T$$

with $\hat{A} := S^H AS$, $\hat{P} := S^H PS$ and $\hat{B} := S^H B$ and is small provided that Z has much fewer columns than n. Hence, it can be solved with direct methods for small scale Lyapunov equations [2], [20], [33], [9] and the improved LRCF is obtained by $Z_+ = Z\hat{R}$, where $\hat{R}$ is a Cholesky factor of $\hat{P}$. For more information we refer to [10].

2) Column Compression: As the iteration proceeds, the number of columns in Z will increase. This will in turn not only increase the required memory for storing Z but also the computational effort for computing the residual norm (2). One way to keep the LRCF as small as possible is to neglect nearly linearly dependent columns of Z. As in [30, Ch. 4.4.1] this can be achieved by applying, e.g., a rank revealing QR decomposition [12] to Z and using the numerical rank with a prescribed tolerance as truncation criterion. Note that when using the Galerkin projection approach, this column compression can be received at almost no additional costs in the orthogonalization procedure of Z.

D. Available Software Including LRCF-ADI

The "Lyapunov Package" [27] (LyaPack)[1] is a MATLAB® toolbox that provides a variety of algorithms for solving different numerical control theory problems, such as large and sparse Lyapunov and algebraic Riccati equations, model order reduction and linear quadratic optimal control problems. There the LRCF-ADI method is implemented in different versions and forms the backbone for solving the occurring large-scale matrix equations.

"Matrix Equations Sparse Solvers" (M.E.S.S.)[2] is the upcoming successor of LyaPack. It includes several generalized and in various ways improved implementations of the solvers for Lyapunov and algebraic matrix equations. It will also provide algorithms for differential Riccati equations. M.E.S.S. will be available as MATLAB and C-version.

III. Real Formulations

Here we briefly describe two older approaches that deal with complex shift parameters before we present our new one.

A. Previous Approaches

1) Real Formulation of LRCF-ADI: In [5], [26], a real formulation of Algorithm 1 is presented, which concatenates steps associated with a pair of complex conjugate shift parameters into one step. Let for this purpose $\nu_j = \{\mu_j, \mu_{j+1} := \overline{\mu_j}\}$ be such a pair and V_j, V_{j+1} the iterates. For instance, if $j > 2$ and μ_{j-1} was real, the iterates V_j, V_{j+1} can then be constructed via

$$\begin{aligned} V_j &= 2\sqrt{-\operatorname{Re}(\mu_j)}|\mu_j|\tilde{V}_j, \;\; V_{j+1} = 2\sqrt{-\operatorname{Re}(\mu_j)}A\tilde{V}_j, \;\; \text{where} \\ \tilde{V}_j &= \left(A^2 + 2\operatorname{Re}(\mu_j)A + |\mu_j|^2 I_n\right)^{-1}\left((A - \mu_{j-1}I_n)\tilde{V}_{j-1}\right), \end{aligned} \tag{4}$$

and $\tilde{V}_{j-1}$ is real due to the special treatment of complex operations in the previous iterations. A complete algorithm is given by [5, Algorithm 4.] and referred to as LRCF-ADI-R, but unfortunately lacking derivations of the involved formulas. Hence, we include them in the appendix.

This reformulation has the advantage that no complex arithmetic operations are required but the disadvantage that linear systems with matrices of the form $A^2 + 2\operatorname{Re}(\mu_j)A + |\mu_j|^2 I_n$ are encountered. For large scale matrices, A^2 might not be computable in an efficient way and even if, it will not preserve the original sparsity of A such that the application of sparse direct solvers is derailed. Iterative solvers can still be applied since they work with matrix-vector products only, such that the explicit construction of the operator is not needed. However, the condition number can be increased due to the squaring which might deteriorate the efficiency of iterative solvers as well.

[1]Available at http://www.tu-chemnitz.de/sfb393/lyapack/ or http://www.netlib.org/lyapack/.
[2]See http://svncsc.mpi-magdeburg.mpg.de/trac/messtrac/wiki .

Algorithm 2 Augmentation of Z by real block columns

```
Input: LRCF approximation $Z_{j-1} \in \mathbb{R}^{n\times(j-1)m}$, iterates $V_j,\ V_{j+1} \in \mathbb{C}^{n\times m}$ w.r.t. $\nu_j = \{\mu_j,\ \overline{\mu_j}\}$
Output: $Z_{j+1} \in \mathbb{R}^{n\times(j+1)m}$, that is, $Z_{j-1}$ augmented by $2m$ new real columns.
1: for $\ell = 1,\dots,m$ do
2:     Compute (thin) singular value decomposition (SVD)
           $U\Sigma W^T = [\mathrm{Re}\,(V_j(:,\ell)), \mathrm{Im}\,(V_j(:,\ell)), \mathrm{Re}\,(V_{j+1}(:,\ell)), \mathrm{Im}\,(V_{j+1}(:,\ell))] \in \mathbb{R}^{n\times 4}$
3:     Partition $U,\ \Sigma$ w.r.t. nonzero singular values
           $U = \left[u_1,\ u_2,\ u_3,\ u_4\right],\ \Sigma = \mathrm{diag}\,(\sigma_1,\ \sigma_2,\ 0,\ 0)$
4:     New real columns of LRCF
           $\tilde{Z}_j(:,\ell) = \sigma_1 u_1$
           $\tilde{Z}_{j+1}(:,\ell) = \sigma_2 u_2$
5: end for
6: $Z_{j+1} = [Z_{j-1},\ \tilde{Z}_j,\ \tilde{Z}_{j+1}] \in \mathbb{R}^{n\times(j+1)m}$.
```

2) LyaPack Approach: Another approach can be found in the latest implementations of the LRCF-ADI method in LyaPack [27]. Unfortunately, it can not be found anywhere in the literature. The crucial point there is that again for $V_j,\ V_{j+1}$ corresponding to $\nu_j = \{\mu_j, \mu_{j+1} := \overline{\mu_j}\}$, provided that $\mathrm{Re}\,(V_j),\ \ \mathrm{Im}\,(V_j)$ have both full row rank m, it holds

$$\mathrm{rank}\,(G) = 2m,\ \ G := [\mathrm{Re}\,(V_j), \mathrm{Im}\,(V_j), \mathrm{Re}\,(V_{j+1}), \mathrm{Im}\,(V_{j+1})] \in \mathbb{R}^{n\times 4m} \tag{5}$$

which will be derived in the next subsection. In LyaPack, the current LRCF iterate Z is expanded by real block columns using Algorithm 2 which processes the imaginary and real parts of $V_j,\ V_{j+1}$ column wise. Hence, the matrix G used there is of dimension $n\times 4$ and has rank 2. To neglect the linearly dependent part, a thin singular value decomposition (SVD) in Step 2 of Algorithm 2

$$\begin{aligned} G = U\Sigma W^T &= [u_1,\ u_2,\ u_3,\ u_4] \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & 0 & \\ & & & 0 \end{bmatrix} [w_1 \quad w_2 \quad w_3 \quad w_4]^T \\ &= [u_1,\ u_2] \begin{bmatrix} \sigma_1 & \\ & \sigma_2 \end{bmatrix} \begin{bmatrix} w_1^T \\ w_2^T \end{bmatrix} \end{aligned}$$

reveals the blocks corresponding to $\mathrm{range}\,(G)$. Adding G to the current LRCF will introduce the new blocks[3]

$$GG^T = [u_1,\ u_2] \begin{bmatrix} \sigma_1^2 & \\ & \sigma_2^2 \end{bmatrix} [u_1,\ u_2]^T = TT^T$$

with $T = [\sigma_1 u_1,\ \sigma_2 u_2]$ in the approximate solution of (1). This is carried out in Step 4 and ensures that only the columns of U which correspond to the linearly independent part of G are added to the LRCF Z_{j-1}. Obviously, this approach generates a real LRCF Z after termination of Algorithm 1, but uses complex arithmetic and requires the computation of m singular value decompositions of $n\times 4$ matrices for each complex pair ν_j encountered. Depending on the actual size of m this can be relatively cost intensive. The applicability of sparse direct solvers is preserved since the complex linear systems are not changed in contrast to the approach before. In the next part we propose a new way to generate real LRCFs and circumvent these drawbacks, as well as making the linear system with $A + \mu_{j+1}I_n = A + \overline{\mu_j}I_n$ redundant.

[3]We point out that in the original implementation a superfluous SVD $QSZ^T = [w_1,\ w_2]^T[w_1,\ w_2]$ is used to get the new blocks via $[\tilde{Z}_j(:,\ell),\ \tilde{Z}_{j+1}(:,\ell)] = T = [\sigma_1 u_1,\ \sigma_2 u_2]QS$.

B. A New Approach Based on the Interconnection of ADI Iterates

Our new approach exploits (5) by using the correct linear combination of the linearly dependent part of G without using a SVD. The following theorem states that V_{j+1} is explicitly known once V_j has been computed.

Theorem 1. Assuming a proper set of shift parameters, for two subsequent block iterates $V_j,\ V_{j+1}$ of Algorithm 1 related to the pair of complex conjugated shifts $\nu_j = \{\mu_j,\ \mu_{j+1} := \overline{\mu_j}\}$ it holds

$$V_{j+1} = \overline{V_j} + \beta_j \operatorname{Im}(V_j) \tag{6}$$

with $\beta_j := 2\frac{\operatorname{Re}(\mu_j)}{\operatorname{Im}(\mu_j)}$. Furthermore, iterates associated to real shifts are always purely real.

Proof: We split the proof into three cases concerning different possible (sub)sequences of shift parameters in the LRCF-ADI method.

Case 1: At first we consider a subsequence of LRCF-ADI iterates $V_{j-1},\ V_j,\ V_{j+1}$ with $j > 2$ and associated shift parameters $\mu_{j-1},\ \mu_j,\ \mu_{j+1} := \overline{\mu_j}$. Let $\mu_{j-1} \in \mathbb{R}_-$ generate a real iterate $V_{j-1} \in \mathbb{R}^{n\times m}$, i.e., all previous shift parameters are for the moment assumed to be real as well. According to Step 7 of Algorithm 1, the first complex iterate V_j is calculated by

$$\begin{aligned} V_j &= \sqrt{\frac{\operatorname{Re}(\mu_j)}{\mu_{j-1}}}\left(I_n - (\mu_j + \mu_{j-1})(A + \mu_j I_n)^{-1}\right) V_{j-1} \\ &= \gamma_1 (A + \mu_j I_n)^{-1}(A - \mu_{j-1} I_n) V_{j-1} \end{aligned}$$

with $\gamma_1 := \sqrt{\frac{\operatorname{Re}(\mu_j)}{\mu_{j-1}}}$. Equivalently,

$$(A + \mu_j I_n)V_j = \gamma_1 (A - \mu_{j-1} I_n) V_{j-1} =: W_1 \in \mathbb{R}^{n\times m}$$

Now splitting μ_j and V_j into their real and imaginary parts reveals

$$(A + \operatorname{Re}(\mu_j) I_n)\operatorname{Re}(V_j) - \operatorname{Im}(\mu_j)\operatorname{Im}(V_j) + \jmath\left[(A + \operatorname{Re}(\mu_j) I_n)\operatorname{Im}(V_j) + \operatorname{Im}(\mu_j)\operatorname{Re}(V_j)\right] = W_1.$$

Since $\operatorname{Im}(W_1) = 0$, the same has to hold for the imaginary part of the left hand side, and therefore

$$\operatorname{Re}(V_j) = -\frac{1}{\operatorname{Im}(\mu_j)}(A + \operatorname{Re}(\mu_j) I_n)\operatorname{Im}(V_j). \tag{7}$$

The next iterate V_{j+1} corresponding to $\overline{\mu_j}$ is constructed via

$$\begin{aligned} V_{j+1} &= V_j - 2\overline{\mu_j}(A + \overline{\mu_j} I_n)^{-1} V_j \\ &= \operatorname{Re}(V_j) + \jmath \operatorname{Im}(V_j) - 2\overline{\mu_j}\left(A + \overline{\mu_j} I_n\right)^{-1}(\operatorname{Re}(V_j) + \jmath \operatorname{Im}(V_j)) \\ &= \operatorname{Re}(V_j) + \jmath \operatorname{Im}(V_j) - 2\overline{\mu_j}(A + \overline{\mu_j} I_n)^{-1}\left(-\frac{1}{\operatorname{Im}(\mu_j)}(A + \operatorname{Re}(\mu_j) I_n)\operatorname{Im}(V_j) + \jmath \operatorname{Im}(V_j)\right) \\ &= \operatorname{Re}(V_j) + \jmath \operatorname{Im}(V_j) + \frac{2\overline{\mu_j}}{\operatorname{Im}(\mu_j)}(A + \overline{\mu_j} I_n)^{-1}\Big((A + \operatorname{Re}(\mu_j) I_n)\operatorname{Im}(V_j) - \jmath \operatorname{Im}(\mu_j)\operatorname{Im}(V_j)\Big) \\ &= \operatorname{Re}(V_j) + \jmath \operatorname{Im}(V_j) + \frac{2\overline{\mu_j}}{\operatorname{Im}(\mu_j)}(A + \overline{\mu_j} I_n)^{-1}(A + \overline{\mu_j} I_n)\operatorname{Im}(V_j) \\ &= \operatorname{Re}(V_j) + \jmath \operatorname{Im}(V_j) + 2\left(\frac{\operatorname{Re}(\mu_j)}{\operatorname{Im}(\mu_j)} - \jmath\right)\operatorname{Im}(V_j) \\ &= \overline{V_j} + \beta_j \operatorname{Im}(V_j) \end{aligned}$$

with $\beta_j := 2\frac{\operatorname{Re}(\mu_j)}{\operatorname{Im}(\mu_j)}$, which is the desired result for Case 1. Note that since B in (1) is real, it is not hard to show that (7) and thus (6) hold also for the first two steps ($j = 1, 2$) of Algorithm 1 if the first shift parameter μ_1 is complex and $\mu_2 = \overline{\mu_1}$.

Case 2: Now we investigate what happens with the next iterates $V_{j+2},\ V_{j+3}$ after Case 1, if the iteration is continued with another complex pair $\mu_{j+2},\ \mu_{j+3} := \overline{\mu_{j+2}}$ of shift parameters. The equation for V_{j+2} is then given by

$$\begin{aligned} V_{j+2} &= \sqrt{\frac{\operatorname{Re}(\mu_{j+2})}{\operatorname{Re}(\mu_{j+1})}}\left(I_n - (\mu_{j+2} + \overline{\mu_{j+1}})(A + \mu_{j+2} I_n)^{-1}\right) V_{j+1} \\ &= \gamma_2 (A + \mu_{j+2} I_n)^{-1}(A - \overline{\mu_{j+1}} I_n) V_{j+1}, \end{aligned}$$

where $\gamma_2 := \sqrt{\frac{\operatorname{Re}(\mu_{j+2})}{\operatorname{Re}(\mu_{j+1})}}$. This is equivalent to

$$(A + \mu_{j+2} I_n) V_{j+2} = \gamma_2 (A - \mu_j I_n) V_{j+1} =: W_2 \in \mathbb{C}^{n \times m}.$$

Since V_{j+1} and V_j were constructed using $\mu_{j+1} := \overline{\mu_j}$ and, respectively, μ_j as shift parameters, we use that (6), (7) hold for V_j and V_{j+1}. Partitioning μ_j and V_{j+1} into their real and imaginary parts in the right hand side W_2 together with (6) leads to

$$\begin{aligned} W_2 =& \gamma_2 \big(A - \operatorname{Re}(\mu_j) I_n - \jmath \operatorname{Im}(\mu_j) I_n\big)\big(\operatorname{Re}(V_{j+1}) + \jmath \operatorname{Im}(V_{j+1})\big) \\ =& \gamma_2 \big(A - \operatorname{Re}(\mu_j) I_n - \jmath \operatorname{Im}(\mu_j) I_n\big)\left(\operatorname{Re}(V_j) + \frac{2\operatorname{Re}(\mu_j)}{\operatorname{Im}(\mu_j)} \operatorname{Im}(V_j) - \jmath \operatorname{Im}(V_j)\right) \\ =& \gamma_2 \left(\left[\big(A - \operatorname{Re}(\mu_j) I_n\big)\left(\operatorname{Re}(V_j) + \frac{2\operatorname{Re}(\mu_j)}{\operatorname{Im}(\mu_j)} \operatorname{Im}(V_j)\right) - \operatorname{Im}(\mu_j)\operatorname{Im}(V_j)\right]\right. \\ & \left. - \jmath \left[\big(A - \operatorname{Re}(\mu_j) I_n\big)\operatorname{Im}(V_j) + \operatorname{Im}(\mu_j)\operatorname{Re}(V_j) + 2\operatorname{Re}(\mu_j)\operatorname{Im}(V_j)\right]\right). \end{aligned}$$

Using (7) yields

$$\operatorname{Im}(W_2) = -\Big(\big(A + \operatorname{Re}(\mu_j) I_n\big)\operatorname{Im}(V_j) + \operatorname{Im}(\mu_j)\operatorname{Re}(V_j)\Big) = 0,$$

which leads to the same situation as in the case before and it follows that (7) holds also for V_{j+2} and μ_{j+2} and thus (6) holds for V_{j+3}. Consequently, if the next shift parameters come also in complex conjugate pairs, (6) holds for $V_{j+(2k+1)}$, $k = 1, 2 \ldots$ which yields the statement of the theorem for this case.

Case 3: Finally, we investigate the situation when after one (or several) pair(s) of complex conjugate shift parameters a real shift is introduced. Let μ_{j+2} be this real shift following a pair of complex conjugate shifts $\mu_j,\ \mu_{j+1} := \overline{\mu_j}$. Then, using similar manipulations as in Case 2 and exploiting (6), (7) for V_{j+1}, one finds

$$(A + \mu_{j+2} I_n)\left(\operatorname{Re}(V_{j+2}) + \jmath \operatorname{Im}(V_{j+2})\right) = W_2 \in \mathbb{R}^{n \times m}$$

which reveals that V_{j+2} is purely real in this situation, regardless if the shift μ_{j+1} before was a complex or a real one. This is exactly the second statement of the theorem. Note that using a complex conjugate pair of shift parameters after Case 3 will lead us back to Case 1. This completes the proof, because we have now covered all possible situations in Algorithm 1. ■

The theorem reveals that in the case of a pair of complex conjugate shifts, an iterate V_{j+1} corresponding to a shift $\mu_{j+1} := \overline{\mu_j}$ can be constructed entirely from the previous iterate V_j and shift μ_j without solving a second complex linear system with $A + \overline{\mu_j} I_n$, which reduces the costs for generating both iterates roughly by one half. However, it is still required to solve one complex linear system.

Relation (6) shows moreover that for two subsequent iterates $V_j,\ V_{j+1}$ generated with a complex pair $\mu_j,\ \overline{\mu_j}$ of shift parameters, statement (5) holds indeed, as it is also implicitly exploited in LyaPack using a singular value decomposition, see Section III-A2. Hence, instead of adding $V_j,\ V_{j+1}$ to the current LRCF, it is sufficient to add $\operatorname{Re}(V_j),\ \operatorname{Im}(V_j)$ in the correct way as we will show next. This will also drastically reduce the amount of memory required to store the complex data.

Let $\hat{Z}$ denote the $n \times 2m$ block to be added to the current LRCF after computing V_j with an LRCF-ADI step and V_{j+1} using (6). Then

$$\hat{Z} = \begin{bmatrix} V_j & V_{j+1} \end{bmatrix} = \begin{bmatrix} \operatorname{Re}(V_j) & \operatorname{Im}(V_j) \end{bmatrix} \hat{T}, \quad \hat{T} := \begin{bmatrix} I_m & I_m \\ \jmath I_m & (\beta_j - \jmath) I_m \end{bmatrix}.$$

With $\tilde{Z} := \begin{bmatrix}\operatorname{Re}(V_j) & \operatorname{Im}(V_j)\end{bmatrix} \in \mathbb{R}^{n\times 2m}$ the contribution of Z to the low-rank solution of (1) is given by

$$\hat{Z}\hat{Z}^H = \tilde{Z}\hat{T}\hat{T}^H\tilde{Z}^T = \tilde{Z}\begin{bmatrix}2I_m & \beta_j I_m\\ \beta_j I_m & (\beta_j^2+2)I_m\end{bmatrix}\tilde{Z}^T.$$

The $2m\times 2m$ matrix $\hat{T}\hat{T}^H$ in the middle is always real symmetric and positive definite which can be shown using

$$\begin{bmatrix}2I_m & \beta_j I_m\\ \beta_j I_m & (\beta_j^2+2)I_m\end{bmatrix} = \hat{F}\otimes I_m, \tag{8}$$

with

$$F := \begin{bmatrix}2 & \beta_j\\ \beta_j & (\beta_j^2+2)\end{bmatrix} \in \mathbb{R}^{2\times 2}.$$

The symmetric matrix F is, due to its construction, at least positive semidefinite and its eigenvalues are

$$\lambda_{1,2}(F) = \frac{1}{2}(\beta_j^2+4) \pm \frac{1}{2}\sqrt{\beta_j^4+4\beta_j^2}.$$

Since $\beta_j \in \mathbb{R}$ and therefore, $(\beta_j^2+4)^2 > \beta_j^4+4\beta_j^2$ holds, $\lambda_{1,2}(F)$ are both strictly positive which provides the positive definiteness of F. Using the spectral property of Kronecker products gives the positive definiteness of the $2m\times 2m$ matrix in (8). Thus, it has a unique Cholesky factorization given by

$$\hat{T}\hat{T}^H = LL^T, \quad L := \frac{\sqrt{2}}{2}\begin{bmatrix}2I_m & 0\\ \beta_j I_m & \sqrt{\beta_j^2+4}\cdot I_m\end{bmatrix} \in \mathbb{R}^{2m\times 2m}.$$

Since

$$\hat{Z}\hat{Z}^H = \tilde{Z}LL^T\tilde{Z}^T = \check{Z}\check{Z}^T, \quad \check{Z} := \tilde{Z}L, \tag{9}$$

we can add the real block $\check{Z} \in \mathbb{R}^{n\times 2m}$ as new block to the LRCF iterate and hence generate a purely real LRCF throughout the whole iteration. The resulting LRCF-ADI using this strategy is shown in Algorithm 3, where storing complex data is only needed temporarily for the iterate V_j and the linear system $A+\mu_j I$ (e.g. for the sparse LU factors) associated to a complex shift μ_j. Compared to the standard LRCF-ADI (Algorithm 1) the memory required for storing the LRCF Z is consequently reduced by a factor of two. Note that adding $\operatorname{Re}(V_j)$ if $\mu_j \in \mathbb{R}_-$ in Step 9, although V_j is by Theorem 1 purely real in this case, ensures that no erroneous imaginary parts caused by rounding errors are added to Z_{j-1}. As shown in Step 14, it is not necessary to form L and compute the product with $\tilde{Z}L$ in (9) explicitly. Furthermore, in order to stay as much in real arithmetics as possible, stopping criteria should be computed only after the complete complex pair is processed. Alternatively, if one is interested in the intermediate residual (or relative change of the LRCF), it is advised to compute it using the appropriate new real blocks $\left[Z_{j-1},\ \sqrt{2}\operatorname{Re}(V_j) + \frac{\beta}{\sqrt{2}}\operatorname{Im}(V_j)\right]$, since $[Z_{j-1},\ V_j]$ will include complex data.

IV. Generalizations

A. *Generalized Lyapunov Equation*

The LRCF-ADI method can be easily generalized to solve large scale *generalized Lyapunov equations*

$$APE^T + EPA^T = -BB^T \tag{10}$$

with an additional sparse matrix $E \in \mathbb{R}^{n\times n}$. For now we assume that E is regular and $\Lambda(A,\ E) \subset \mathbb{C}_-$ which ensures the existence of the unique solution P. In this case we can formally multiply (10) from the left by E^{-1} and from the right by E^{-T} to get a Lyapunov equation of the form (1) with $N := E^{-1}A$ and $G := E^{-1}B$ such that Theorem 1 and Algorithm 3 can be applied. However, following the manipulations in [3],[30, Ch. 5.2.2.] shows that the iteration steps can be rewritten to

$$\begin{aligned}V_1 &= \sqrt{-2\operatorname{Re}(\mu_1)}(A+\mu_1 E)^{-1}B,\\ V_j &= \sqrt{\operatorname{Re}(\mu_j)/\operatorname{Re}(\mu_{j-1})}(V_{i-1} - (\mu_j + \overline{\mu_{j-1}})(A+\mu_j E)^{-1}EV_{i-1}),\ j\geq 2.\end{aligned}$$

Algorithm 3 LRCF-ADI with realification

```
Input: $A$ and $B$ as in (1) and shift parameters $\{\mu_1, \dots, \mu_{j_{max}}\}$.
Output: $Z = Z_{jmax} \in \mathbb{R}^{n \times t j_{max}}$, such that $ZZ^T \approx P$
 1: for $j = 1, 2, \dots, j_{max}$ do
 2:   if $j = 1$ then
 3:     Solve $(A + \mu_1 I_n)V_1 = \sqrt{-2\,\mathrm{Re}\,(\mu_1)}B$ for $V_1$.
 4:   else
 5:     Solve $(A + \mu_j I_n)\tilde{V} = V_{j-1}$ for $\tilde{V}$.
 6:     $V_j = \sqrt{\mathrm{Re}\,(\mu_j)/\,\mathrm{Re}\,(\mu_{j-1})}(V_{j-1} - (\mu_j + \overline{\mu_{j-1}})\tilde{V})$.
 7:   end if
 8:   if $\mathrm{Im}\,(\mu_j) = 0$ then
 9:     $V_j = \mathrm{Re}\,(V_j)$.
10:     $Z_j = [Z_{j-1},\ V_j]$.
11:   else
12:     $\beta = 2\frac{\mathrm{Re}\,(\mu_j)}{\mathrm{Im}\,(\mu_j)}$.
13:     $V_{j+1} = \overline{V_j} + \beta\,\mathrm{Im}\,(V_j)$.
14:     $Z_{j+1} = \left[Z_{j-1},\ \sqrt{2}\,\mathrm{Re}\,(V_j) + \frac{\beta}{\sqrt{2}}\,\mathrm{Im}\,(V_j),\ \sqrt{\frac{\beta^2}{2} + 2} \cdot \mathrm{Im}\,(V_j)\right]$.
15:     Set $j = j + 1$.
16:   end if
17: end for
```

Inserting these formulas into Algorithm 1 leads to the generalized LRCF-ADI method (G-LRCF-ADI) [30], [11], [3] which can be rewritten, using the new realification approach of the previous section, to a generalized version of Algorithm 3. Of course, the other two strategies are applicable as well, but using the complete real formulation of G-LRCF-ADI will result in linear systems involving the matrix

$$AE^{-1}A + 2\,\mathrm{Re}\,(\mu)A + |\mu|^2 E. \tag{11}$$

Since applying the inverse of E to A is not feasible in a large-scale setting this approach is rendered less preferable.

A similar argumentation holds if $A,\ E$ and B are structured, e.g., in the form

$$E := \begin{bmatrix} I_n & 0 \\ 0 & M \end{bmatrix},\ A := \begin{bmatrix} 0 & I_n \\ -K & -D \end{bmatrix} \in \mathbb{R}^{2n \times 2n},\ B := \begin{bmatrix} 0 \\ B_1 \end{bmatrix} \in \mathbb{R}^{2n \times m}, \tag{12a}$$

or

$$E := \begin{bmatrix} D & M \\ M & 0 \end{bmatrix},\ A := \begin{bmatrix} -K & 0 \\ 0 & M \end{bmatrix} \in \mathbb{R}^{2n \times 2n},\ B := \begin{bmatrix} B_1 \\ 0 \end{bmatrix} \in \mathbb{R}^{2n \times m}, \tag{12b}$$

where $M,\ D,\ K \in \mathbb{R}^{n \times n}$ are nonsingular matrices and $B_1 \in \mathbb{R}^{n \times m}$. This is especially the case when $M,\ D,\ K,\ B_1$ define a second order linear time invariant dynamical system [11] and $E,\ A,\ B$ represent a transformation to a first order generalized system. Note that there are other possible choices to carry out this transformation [34], but all of those can be dealt with similarly. By assuming again that $\Lambda(A,\ E) \subset \mathbb{C}_-$ and by exploiting the structure of $M,\ D,\ K,\ B$, it is possible to solve (10) by a version of G-LRCF-ADI which works directly using the $n \times n$ matrices $M,\ D,\ K$ and the right hand side B_1. This modified version is then called second order LRCF-ADI (SO-LRCF-ADI) [30], [11] and can certainly be equipped with the realification strategy, too.

Now let the matrices defining (10) be of the structure

$$E = \begin{bmatrix} E_1 & 0 \\ 0 & 0 \end{bmatrix},\ A = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix},\ B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}. \tag{13}$$

with $E_1 \in \mathbb{R}^{n_f \times n_f}$ and $A_4 \in \mathbb{R}^{(n-n_f)\times(n-n_f)}$ nonsingular. All the other subordinate block matrices have appropriate dimensions. The number $n_f < n$ refers to the number of finite eigenvalues in $\Lambda(A,\ E)$, which are supposed to be located in $\mathbb{C}_-$ as in all cases before. Note that the special structure of E implies that the pair $(A,\ E)$ is of index one, i.e., there are no Jordan chains of length greater than one belonging to the infinite eigenvalues. Since the inverse of E does not exist, the formulation of an associated Lyapunov equation and G-LRCF-ADI method is not as straightforward as before. In [16] an approach is presented which works on the generalized Lyapunov equation $\hat{A}\hat{P}\hat{E}^T + \hat{E}\hat{P}\hat{A}^T = -\hat{B}\hat{B}^T$ with

$$\hat{E} := E_1,\ \tilde{A} := A_1 - A_2A_4^{-1}A_3 \in \mathbb{R}^{n_f\times n_f} \tag{14}$$

and the right hand side factor $\hat{B} := B_1 - A_2A_4^{-1}B_2 \in \mathbb{R}^{n_f\times m}$. Solving this generalized Lyapunov equation with G-LRCF-ADI of course allows the use of the new realification approach, too. However $\hat{A}$ will in general be a dense matrix which prohibits the use of sparse solvers for the linear systems, for example, $(\hat{A} + \mu_1\hat{E})V_1 = \hat{B}$ in the first iteration. The main idea in [16] is to solve V_1 from the sparse linear system of equations

$$\begin{bmatrix} A_1 + \mu_1 E_1 & A_2 \\ A_3 & A_4 \end{bmatrix}\begin{bmatrix} V_1 \\ \Gamma \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix},$$

where $\Gamma \in \mathbb{R}^{(n-n_f)\times m}$ is an auxiliary variable of no further use. For all other iterations the right hand side is $[V_{j-1}^T E_1^T,\ 0^T]^T$. These augmented linear systems can now be treated efficiently by sparse techniques, and the resulting modification of the LRCF-ADI is called sparse LRCF-ADI (SLRCF-ADI). Furthermore, they do not hinder the application of our novel realification approach since the method still works and continues only with the V_j blocks. In [15] it is shown that this framework can be generalized to a special class of index-2 matrix pairs resulting from an application concerning RLC circuit models.

Remark. Note that it is claimed in [16] that the columns in the LRCF Z which where generated using μ_j and $\overline{\mu_j}$ are complex conjugate to each other and hence, Z can be transformed into a real form by applying

$$T = \frac{\sqrt{2}}{2}\begin{bmatrix} 1 & \jmath \\ 1 & -\jmath \end{bmatrix}$$

to these columns. As Theorem 1 shows, only the imaginary parts of these columns are complex conjugate versions of each other, rendering the above claim and the transformation incorrect.

For matrix pairs of arbitrary index, i.e., the infinite eigenvalues have Jordan chains of length greater than one, the authors in [24] propose to solve *generalized projected Lyapunov equations*

$$APE^T + EPA^T = -\Pi_\ell BB^T\Pi_\ell^T,\ P = \Pi_r P \Pi_r^T,$$

where $\Pi_\ell, \Pi_r \in \mathbb{R}^{n\times n}$ are spectral projectors onto the left and right deflating subspaces of $(A,\ E)$. With $P \approx ZZ^T$ as before, these equations can again be solved by G-LRCF-ADI using the projected right hand side $\Pi_\ell B \in \mathbb{R}^{n\times m}$, interchanging the roles of A and E, and using the reciprocals $\tau_j = \mu_j^{-1}$ of the shift parameters:

$$\begin{aligned} Z_1 &= \sqrt{-2\operatorname{Re}(\tau_1)}(E + \tau_1 A)^{-1}\Pi_\ell B, \\ Z_j &= \sqrt{\frac{\operatorname{Re}(\tau_j)}{\operatorname{Re}(\tau_{j-1})}}\left(I_n - (\tau_j + \overline{\tau_{j-1}})(E + \tau_j A)^{-1}Z_{j-1}\right). \end{aligned}$$

Since all other assumptions remain, the new realification approach as in Algorithm 3 is also applicable here.

B. Algebraic and Differential Riccati Equations

In this section we briefly review *algebraic Riccati equations* (AREs) and the low-rank Newton method for their solution as we will later test the performance of the realification strategy within this approach. An ARE is given by

$$A^TX + XA - XBR^{-1}B^TX + Q = 0 \tag{15}$$

with $A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m}$ similar to (1), $R \in \mathbb{R}^{m\times m}$ positive definite and $Q \in \mathbb{R}^{n\times n}$ positive semidefinite. Equations of the form (15) play an important role in optimal control problems for linear time invariant control systems, where the right hand side matrix is often given as $Q := C^T\hat{Q}C$ with $C \in \mathbb{R}^{p\times n}$ and $\hat{Q} = \tilde{Q}\tilde{Q}^T \in \mathbb{R}^{p\times p}$ positive definite. There, a *stabilizing solution* of (15), or respectively a stabilizing *feedback matrix* $G := XBR^{-1}$

Algorithm 4 Low-Rank Cholesky Factor Newton Method (LRCF-NM) for AREs

```
Input: $A,\ B,\ Q = C^T\tilde{Q}^T\tilde{Q}C,\ R = \tilde{R}\tilde{R}^T$ as in (15) and initial stabilizing feedback $G_0$.
Output: Stabilizing solution $X$ of (15) and feedback matrix $G = XBR^{-1}$.
1: for $k = 1, 2, \dots$ do
2:   Compute ADI shift parameters for $A^T - G_{k-1}B^T$.
3:   Set $L_k = [C^T\tilde{Q},\ G_{k-1}\tilde{R}]$.
4:   Solve
         $(A^T - G_{k-1}B^T)X_k + X_k(A - BG_{k-1}{}^T) = -L_kL_k^T$
     for a LRCF $Z_kZ_k^H \approx X_k$, e.g., with Algorithm 1 or 3.
5:   Update feedback matrix $G_k = Z_k\left(Z_k^HBR^{-1}\right)$.
6: end for
```

is sought, such that $A - BG^T$ is Hurwitz. Under the present properties of the involved matrices the stabilizing solution is unique and symmetric if in addition $(A,\ BR^{-1}B^T)$ is *stabilizable* and $(A,\ Q)$ is *detectable*. One widely used approach to solve AREs is using a Newton method which will require the solution of a Lyapunov equation in every iteration step. The Newton-Kleinman iteration for AREs [21] is one version of such methods. Displayed in Algorithm 4 is the Low-Rank Cholesky Factor Newton Method which exploits the special structure of the inner Lyapunov equations. By assuming $p,\ m \ll n$ it is possible to solve the Lyapunov equation in Step 4 for a LRCF Z_k such that $Z_k{Z_k}^T \approx X_k$. This can be carried out with LRCF-ADI and thus, with our modification given in Algorithm 3 concerning complex shift parameters. Of course, the other two strategies are also applicable here. The shift parameters are now related to the matrix $A^T - G_{k-1}B^T$ and may be updated in each iteration. Since this matrix will in general be dense but is decomposed into a sparse part and a low-rank term, it is advised to solve the shifted linear systems involving $A^T - G_{k-1}B^T + \mu_jI_n$ with the help of the *Sherman-Morrison-Woodbury formula* (e.g. [18]). This is still the case for the completely real formulation as in Section III-A1. The matrix of the double step corresponding to a complex pair μ can be rewritten into (skipping the inner and outer iteration indices j and k)

$$\left(A^T - GB^T + \mu I_n\right)\left(A^T - GB^T + \overline{\mu}I_n\right) = A_S - UW^T$$

with

$$A_S := {A^T}^2 + 2\,\mathrm{Re}\,(\mu)A^T + |\mu|^2I_n, \quad U := \left[A^TG,\ G,\ G\right], \quad W := \left[B,\ AB,\ \left(2\,\mathrm{Re}\,(\mu)B - B(G^TB)\right)\right].$$

The matrix A_S is sparse and $U,\ W \in \mathbb{R}^{n\times 3m}$ are of low-rank, such that Sherman-Morrison-Woodbury still applies.

It is furthermore possible to rewrite Algorithm 4 to take the low-rank structure of X_k fully into account and avoid the explicit construction of $Z_kZ_k^H$ which leads to the Implicit Low-Rank Cholesky Factor Newton Method (LRCF-NM-I). See, for instance, [30], [1], [27], [5] for details.

Algorithm 4 can, for instance, be terminated if the normalized ARE residual

$$\|A^TX_k + X_kA - X_kBR^{-1}B^TX_k + C^TC\|_2\ /\ \|C^TC\|_2 < \epsilon_{\text{ARE}}, \tag{16}$$

or if the relative change of the ARE's LRCF

$$\frac{\|Z_kZ_k^H - Z_{k-1}Z_{k-1}^H\|_2}{\|Z_kZ_k^H\|_2} \tag{17}$$

drops below a prescribed tolerance, as well. The relative change of the computed feedback matrix can also be used as stopping criterion which is especially important for LRCF-NM-I.

Acceleration techniques via Galerkin projections similar to the one in LRCF-ADI for Lyapunov equations are also applicable. The projection can be carried out to the ARE to be solved as well as in the LRCF-ADI iterations for the inner Lyapunov equations [10].

All these Newton style methods extend in a straightforward way to generalized AREs

$$A^TXE + E^TXA - E^TXBR^{-1}B^TXE + Q = 0.$$

TABLE I
THE PARAMETERS FOR THE LRCF-ADI METHOD AND ITS GENERALIZATIONS.

Parameter	Meaning
k_+	number of Ritz values of A (or $E^{-1}A$)
k_-	number of Ritz values of A^{-1} (or $A^{-1}E$)
$J = J_{\mathbb{R}} + 2J_{\mathbb{C}}$	number of shift parameters (real ones plus complex pairs)
$j_{\max}$	maximal number of ADI iterations
$k_{\max}$	maximal number of (outer) Newton iterations
ϵ_{res}	tolerance for normalized Lyapunov residual (2)
ϵ_{ARE}	tolerance for the normalized ARE residual
ϵ_{rc}	tolerance for relative change (3) of LRCF

Differential Riccati equations (DREs)

$$\dot{X}(t) = A^T X(t) + X(t)A - X(t)BR^{-1}B^T X(t) + Q \tag{18}$$

are also an important tool in control theory. Recent developments concerning their solution in the large-scale case use, e.g., BDF methods [7] which require the solution of an ARE in each step, or Rosenbrock type methods [8] involving the solution of a Lyapunov equation in each step. Both ways involve either implicitly or explicitly large-scale Lyapunov equation, such that LRCF-ADI can be applied. Hence, if complex parameters are involved, they can also benefit from our novel approach for generating real LRCF.

V. NUMERICAL EXAMPLES

In this section we test the different ways to generate real LRCFs with the LRCF-ADI method. We employ the following methods

M1 LRCF-ADI (Algorithm 1),
M2 LRCF-ADI-R (cf. Section III-A1, [5, Algorithm 4.]),
M3 LRCF-ADI with LyaPack realification (Algorithm 2 in Section III-A2),
M4 novel approach given in Algorithm 3 in Section III-B

for solving standard (1) and generalized (10) Lyapunov equations, where we used the associated generalized versions of the above algorithms for the latter case. We also test the different approaches with matrices having the structures (12) and (13) using SO-LRCF-ADI and SLRCF-ADI, respectively, although we do not consider the M2 version there. Later on we briefly investigate the use of the different realification approaches within LRCF-NM (Algorithm 4) to solve AREs.

Table I summarizes the parameters needed for one run of LRCF-ADI, including the ones for the heuristic shift parameter computation. Most oft the parameters are equally used in the generalizations of LRCF-ADI, where others only play a role for a Newton style method to solve AREs. Note that in the remainder $J_{\mathbb{R}}$ denotes the number of real shifts, and respectively $J_{\mathbb{C}}$ the number of pairs of complex conjugate shifts.

All experiments have been carried out in MATLAB 7.11.0 on an Intel®Xeon®W3503 CPU with 2.40 GHz and 6 GB RAM. If not stated otherwise, the occurring linear systems were solved with the MATLAB backslash operator which employs sparse direct techniques.

A. *Standard and Generalized Lyapunov Equations*

We consider the following test examples, starting with two standard Lyapunov equations (1).

Example 1. A standard Lyapunov equation, where the matrix A comes from a finite difference discretization of the parabolic partial differential equation

$$\frac{dx}{dt} = \triangle x - 10\xi_1 \frac{\partial x}{\partial \xi_1} - 1000\xi_2 \frac{\partial x}{\partial \xi_2} \quad \text{on} \quad \Omega = (0,1)^2,$$

where $x = x(\xi_1,\ \xi_2,\ t)$ and homogeneous Dirichlet boundary conditions are imposed. Using 50 equidistant grid points for each spatial dimension results in a stable and sparse matrix $A \in \mathbb{R}^{n\times n}$ with $n = 2500$. The matrix B is chosen as random vector of length n, i.e., $m = 1$. Using these settings this example is very close to [26, Example 6.3.] and to `example_l1.m` in LyaPack.

Example 2. The next example is a 3-d variant of the previous example from [31] with $n = 10648$ and $m = 10$.

The next two text examples represent generalized Lyapunov equations (10) and are part of the Oberwolfach Model Reduction Benchmark collection[4]. Both models belong the the Oberwolfach Benchmark ID 38867.

Example 3. The coefficient matrices with $n = 9669,\ m = 1$ of this generalized Lyapunov equation come from a spatial finite element discretization of the partial differential equation describing the convective heat flow in a two-dimensional anemometer like structure involving both a solid body and a liquid fluid.

Example 4. A similar example as above but now the underlying structure is modeled in three spatial dimensions and represents a chip cooled by a convective flow. The dimensions of the resulting generalized Lyapunov equation are $n = 20082$ and $m = 1$.

We continue with two generalized problems where the matrices stem from a linearization of a second order dynamical system and have a structure of the form (12b).

Example 5. The scalable triple chain oscillator [35] describes three coupled chains of masses interlinked with springs and dampers. The mass and stiffness matrices M and K are symmetric and of dimension $n = 1501$, whereas the damping matrix D is modeled as $D = \alpha_1 M + \alpha_2 D$, $\alpha_1 = 0.02,\ \alpha_2 = 0.05$ and B is a random matrix of dimension $\mathbb{R}^{n\times 5}$. The resulting generalized Lyapunov equation has then a dimension of 3002.

Example 6. The Butterfly Gyro is also taken from the Oberwolfach Collection[3] and models a vibrating mechanical gyro for the use in an inertia sensor. The original matrices arise from a finite element discretization and are of dimension $n = 17361$. The damping matrix is $D = \alpha_1 M + \alpha_2 D$, $\alpha_1 = 10^{-5}$, $\alpha_2 = 10^{-6}$. The equivalent first order matrices are of dimension 34722. The system comes with an output matrix $C \in \mathbb{R}^{12\times n}$ and we use $B = [C, 0]^T$ in the right hand side.

The Brazilian interconnected power system models[5] provide a number of test-systems which have the structure (13) with $E_1 = I_{n_f}$ (see also [16] for more information). The corresponding Lyapunov equations can be solved with SLRCF-ADI.

Example 7. As first example of this class we choose system bips98_606 which has dimension $n = 7135$, where $n_f = 606$ and $m = 4$.

Example 8. System bips07_3078 with $n = 21128$, $n_f = 3078$ and $m = 4$.

All four variants of (G-)LRCF-ADI are tested with Examples 1 – 4. The normalized (generalized) residual norm (2) is computed in every step and used as stopping criterion. The norms of the residual are computed with a Lanczos process which is initialized by a random vector. The heuristic shift parameters are computed with two Arnoldi processes using $B\hat{e}$ as starting vector, where $\hat{e} := (1, \dots, 1)^T \in \mathbb{R}^m$.

For the first example Figure 1 illustrates the normalized residual norm against the iteration number, where we also included one run of LRCF-ADI using only the real parts of the shift parameters used. The four runs using also complex shift parameters show almost the same residual norms during the iteration and converge after 98 iterations to the desired accuracy of $\epsilon_{\text{res}} = 10^{-10}$. This illustrates the proven mathematical equivalence of the different approaches. The small deviations in the residuals occur exactly at the iterations which work with a pair of complex shifts, since the intermediate residual after the first complex shift parameter consists of a slightly different new block in the LRCF for each variant. After the conjugate shift has been processed, too, the residual norms coincide again.

In this example we also like to emphasize the need for complex shift parameters in the unsymmetric case. As it can be clearly observed in the corresponding curve in Figure 1, using only the real parts of the shift parameters seems to slow down the convergence speed remarkably.

[4]Available at http://portal.uni-freiburg.de/imteksimulation/downloads/benchmark.

[5]Available at http://sites.google.com/site/rommes/software.

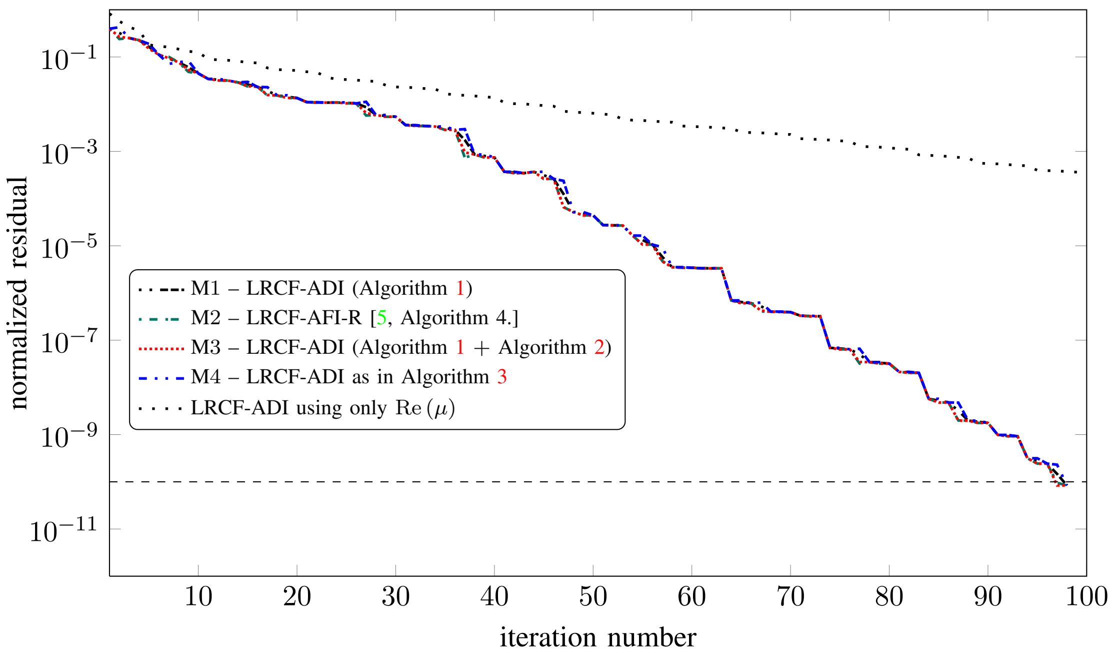

Fig. 1. History of the normalized residual norms obtained with LRCF-ADI and its variants generating real LRCFs for Example 1. The dashed thin black line indicates the chosen tolerance ϵ_{res}.

Now we compare the different variants in terms of the computation times needed until termination. The results and the used parameters are summarized in Table II. For Example 1 the novel approach (M4) needs less time than the other versions, where the run of LRCF-ADI without any special treatment of complex shift parameters (M1) has not surprisingly the largest computation time followed by the SVD-based approach (M3). However, (M4) is only slightly better compared to the completely real variant LRCF-ADI-R (M2) since the matrix in this example is multidiagonal with a low bandwidth, such that forming A^2 does not take much computational effort and neither introduces much fill-in.

All four version perform similarly for the next example, but here M4 outperforms the other approaches significantly. It needs approximately 55 percent of the computation time of both M2 and M3, and respectively 45 per cent of the time of M1. Despite the larger dimensions of this example and the presence of $m = 10$ right hand sides in the linear systems, there are also other reasons for this greater difference compared to the example before. On the one hand, A has a greater bandwidth and thus, solving the two shifted linear systems in M2, M3 or the one involving A^2 in M2 increases the computation time by a considerably larger ratio than in Example 1. This can be clarified by comparing the computation times needed for solving the linear system(s) in each method for a pair of complex conjugate shift parameters. Let $\mu, \overline{\mu}$ be an arbitrary pair of complex conjugated numbers from the set of shift parameters. Solving one shifted linear system involving μ and an arbitrary right hand side of appropriate size requires about 1.272 seconds. This is the main computation effort in M4, whose time is roughly doubled in M1 and M3 to 2.51 seconds, where in the latter case the application of Algorithm 2 computing 10 SVDs takes additional 0.022 seconds. For the solution of the linear system with $A^2 + 2\,\mathrm{Re}\,(\mu)A + |\mu|^2 I_n$ approximately 2.49 seconds elapse including 0.013 seconds for forming A^2. Note that this matrix has 246136 nonzero entries in contrast to the original 71632 of A.

Comparable observations can be made for the generalized Lyapunov equations of Examples 3 and 4. All four variants of G-LRCF-ADI converge in the same manner and show the same ranking with respect to the computation time: the novel approach M4 requires the least time, followed by M2, M3 and finally the original LRCF-ADI M1. One exception is approach M2 in Example 3 which did not converge to the desired accuracy within 50 iterations. In fact, the final normalized residual was of order 10^6. One explanation might be that forming $AE^{-1}A$ in (11) induces for this example enough rounding errors to render the overall process inapplicable.

Examples 5 and 6 are treated with the associated formulation SO-LRCF-ADI which takes the present structure into account. We do not consider the approach M2 for these problems. Comparing the runtimes confirms again that

TABLE II
PARAMETERS AND COMPUTATION TIMES FOR EXAMPLES 1 – 6. IF THE TIME IS WRITTEN WITHIN BRACKETS THE METHOD DID NOT CONVERGE WITHIN $j_{\max}$ ITERATIONS. OTHERWISE CONVERGENCE OCCURRED FOR ALL VARIANTS AT j_{ITER} ITERATIONS.

Example	Parameters								Computation time in seconds			
	n	m	k_+	k_-	$J,\ (J_\mathbb{R},\ J_\mathbb{C})$	ϵ_{res}	$j_{\max}$	j_{iter}	M1	M2	M3	M4
1	2500	1	40	20	$10,\ (2,\ 4)$	10^{-10}	100	98	3.942	2.671	3.314	2.122
2	10648	10	60	40	$41,\ (3,\ 19)$	10^{-10}	100	78	143.654	118.539	119.390	65.967
3	9669	1	50	40	$40,\ (26,\ 7)$	10^{-10}	50	37	5.777	(6.575)	5.532	4.057
4	20082	1	20	50	$31,\ (21,\ 5)$	10^{-10}	50	27	88.789	71.542	79.992	61.020
5	1501	5	60	60	$51,\ (9,\ 21)$	10^{-8}	150	146	17.473	–	7.769	7.060
6	17361	12	100	110	$64,\ (12,\ 26)$	10^{-6}	100	50	288.875	–	(460.531)	138.835

TABLE III
PARAMETERS AND COMPUTATION TIMES FOR EXAMPLES 7 AND 8.

Example	Parameters									Computation time in seconds		
	n	n_f	m	k_+	k_-	$J,\ (J_\mathbb{R},\ J_\mathbb{C})$	ϵ_{res}	α	$j_{\max}$	M1	M3	M4
7	7135	606	4	90	60	$70,\ (10,\ 30)$	10^{-8}	0.05	161	14.394	14.676	8.360
8	21128	3078	4	50	60	$45,\ (7,\ 19)$	10^{-6}	0.08	194	43.178	42.599	23.952

using the novel approach is superior in terms of the computation time. Note that for Example 6 all shift parameters with $|\mu| > \frac{1}{\sqrt{\epsilon_{\text{mach}}}}$ were neglected from the originally computed 100 since they lead to severe numerical problems which prevented convergence. This effect might be caused by the $\mu^2 M - \mu D + K$ matrices in SO-LRCF-ADI. If a shift parameter is too large, its square can easily overwhelm any influence of the matrices D and K. Still, the SVD-based variant M3 did not fully achieve the desired accuracy after the maximum number of iterations, but ended in a stagnation phase and a final normalized residual norm of order 10^{-5}.

For the Examples 7 and 8 the SLRCF-ADI method is used including the different strategies, but also without M2. The residual norm is not computed for these examples since this would require matrix-vector multiplications with the matrices $\tilde{E}$, $\tilde{A}$ and $\tilde{B}$ in (14) and therefore involving the solution of a linear system with A_4 in each Lanczos step. The computation of the residual norm would then clearly be the dominant factor in the runtime. Instead we choose to terminate the algorithms after $j_{\max}$ iterations which corresponds to the number of iterations required to achieve a relative residual norm below ϵ_{res} in an experiment where the residual norm was really monitored. The matrices $\tilde{A}$ often involve eigenvalues very close to the imaginary axis. To reduce possible numerical difficulties resulting from nearly unstable eigenvalues, we use – similar to [16, Appendix B] – the shifted matrix $\tilde{A} - \alpha I_{n_f}$ with $\alpha > 0$ instead, such that the spectrum of the used matrix is moved farther into the left half plane. The used parameters and runtimes are summarized in Table III. Note that the chosen values of $k_+,\ k_-,\ J,\ \alpha$ are the same as in the numerical experiments in [16]. Again the novel approach requires significantly less time for the computations than M1 and M3.

To conclude, in all examples LRCF-ADI equipped with the new approach which exploits the interconnection between the iterates associated to complex shifts has a lower runtime than the other strategies dealing with this issue. Similar numerical tests (not reported here) revealed that if applicable ($AE^T + EA^T < 0$) and properly implemented, this runtime can be even further reduced by using a Galerkin projection after a couple of LRCF-ADI steps. This projection, as well as column compression using an RRQR, naturally benefit from real LRCFs which reduce their computational effort, too.

B. *Algebraic Riccati Equations*

Now we test how LRCF-NM performs with the different strategies employed in the inner LRCF-ADI iterations for the inherent Lyapunov equation in each Newton iteration. As test problems we choose the matrices of Example 1 and 2 with $Q = I_n,\ R = I_m$ and $C = B^T$ in both cases. In each Newton iteration new shift parameters with

TABLE IV
PARAMETERS, NUMBER OF NEWTON ITERATIONS, COMPUTATION TIMES AND MAXIMAL NUMBER OF REQUIRED LRCF-ADI ITERATIONS $j^{\max}$ FOR LRCF-NM WITH DIFFERENT REALIFICATION METHODS.

Example	Parameters									Computation time in seconds			
	n	m	k_+	k_-	J	$\approx \epsilon_{\text{ARE}}$	ϵ_{res}	k_{iter}	$j^{\max}$	M1	M2	M3	M4
1	2500	1	50	25	15	$5.5 \cdot 10^{-13}$	10^{-10}	4	86	23.854	14.233	17.257	12.541
2	10648	10	60	40	40	$2.3 \cdot 10^{-12}$	10^{-10}	6	66	1709.134	1231.140	1099.657	658.539

respect to $A - G_{k-1}B^T$ were computed with the heuristic approach using the same values of k_+, k_- and J each time. Both examples involve a stable matrix A and thus the initial stabilizing solution G_0 could be set to zero. We terminate LRCF-NM if the normalized ARE residual (16) or the relative change (17) drops below $\epsilon_{\text{ARE}} := n\epsilon_{\text{mach}}$. The inner LRCF-ADI iteration stops if the relative residual norm (2) drops below 10^{-10}. As before, all 2-norms were computed with a Lanczos process.

Table IV summarizes for each example the parameters used and the computation times achieved together with the maximum number of required ADI iterations. Apparently, the savings with respect to the runtime obtained with M4 for the Lyapunov equations decrease the timings for LRCF-NM as well for both examples. For all variants and for both examples the stopping criterion (17) terminated LRCF-NM after k_{iter} Newton iterations. The histories of (16) and (17) of all four strategies were visually barely distinguishable from each other and are hence omitted. The final obtained normalized residual norm was of order 10^{-11} for both examples.

Not shown in Table IV is that using M1, i.e., by not handling complex parameters at all, resulted in a higher number of the maximum required LRCF-ADI iterations: 96 in Example 1 and, respectively, 72 for Example 2. For an explanation consider Step 5 in Algorithm 4, where the approximate low-rank solution $Z_k Z_k^H$ of the Lyapunov equation enters the new feedback matrix G_k. Without any care of complex shift parameters, rounding errors might induce complex data into $Z_k Z_k^H$ and thus into G_k, even if a proper set of shift parameters is used. Since G_k enters in Step 3 the right hand side L_{k+1} of the Lyapunov equation of the next Newton iteration, this might alter the behavior of the next run(s) of LRCF-ADI. This could also be observed in the examples where these differences in the ADI iterations started occurring with the second Newton iteration. There, due to the zero initial solution, a computed feedback G_k enters for the first time the computations. The very first iteration of LRCF-NM including the run of LRCF-ADI were the same for each variant, neglecting the minor differences in the Lyapunov residual norms with respect to complex shift parameters as described in the Lyapunov examples (see Figure 1).

VI. CONCLUSIONS

We investigated the LRCF-ADI method for large-scale Lyapunov equations involving unsymmetric matrices and thus introducing complex shift parameters. After reviewing two older strategies to handle these complex shifts and generating real low-rank factors, we derived a novel approach which exploits the interconnection of the iterates. The resulting big advantage is that only one linear system has to be solved for each pair of complex conjugate parameters. This decreases the amount of computational work and storage needed roughly by one half. Although a part of the computations still employs complex arithmetics, the obtained LRCF is real and the overall algorithm is superior in terms of the runtime compared to the other real formulations, as our numerical tests confirm. There, the novel approach was also tested within generalized versions of LRCF-ADI for solving generalized Lyapunov equations of different nature. Furthermore, it was implemented in the LRCF-NM for solving AREs. In both cases the solution of the considered matrix equations required less computation time than with the other approaches. We point out that on architectures which are not able to handle complex data and computations efficiently, the completely real approach is to be preferred.

Future research perspective could include similar investigations concerning low-rank ADI based methods for large-scale Sylvester equations [6], as well as for discrete (generalized) Lyapunov and algebraic Riccati equations [4].

Acknowledgment

The authors are grateful for the helpful discussion with J. Rommes and F. Freitas regarding SLRCF-ADI and for pointing towards the BIPS examples.

Appendix A
Derivation of the complete real LRCF-ADI

Here we present the derivation of the formulas in LRCF-ADI-R which was mentioned in Section III-A1 since these derivations were not shown in the original paper [5].

If the very first shift parameters come as complex conjugate pair, we plug the result of the first iteration of Algorithm 1

$$V_1 = \gamma_1 \left(A + \mu_1 I_n\right)^{-1} B,$$

with $\gamma_1 := \sqrt{-2\,\mathrm{Re}\,(\mu_1)}$ into the second iteration

$$\begin{aligned} V_2 &= (V_1 - (\mu_2 + \overline{\mu_1})(A + \mu_2 I_n)^{-1} V_1) \\ &= (A + \overline{\mu_1} I_n)^{-1}(A - \overline{\mu_1} I_n) V_1 \\ &= \gamma_1 (A + \overline{\mu_1} I_n)^{-1}(A - \overline{\mu_1} I_n)\left(A + \mu_1 I_n\right)^{-1} B \\ &= \gamma_1 (A - \overline{\mu_1} I_n)(A + \overline{\mu_1} I_n)^{-1}\left(A + \mu_1 I_n\right)^{-1} B, \end{aligned}$$

where we used that $(A - \overline{\mu_1} I_n)$ commutes with $(A + \overline{\mu_1} I_n)$. Since $(A \pm \overline{\mu} I_n)\,(A \pm \mu I_n) = A^2 \pm 2\,\mathrm{Re}\,(\mu_1) A + |\mu|^2 I_n$ for all $\mu \in \mathbb{C}$, the above equation becomes

$$\begin{aligned} V_2 &= \gamma_1 (A - \overline{\mu_1} I_n)\left(A^2 + 2\,\mathrm{Re}\,(\mu_1) A + |\mu|^2 I_n\right)^{-1} B \\ &= \gamma_1 \left(-\overline{\mu_1} \tilde{V}_1 + \tilde{V}_2\right) \end{aligned} \tag{19}$$

with $\tilde{V}_1 := \left(A^2 + 2\,\mathrm{Re}\,(\mu_1) A + |\mu|^2 I_n\right)^{-1} B$ and $\tilde{V}_2 := A\tilde{V}_1$. Using these definitions it is not hard to see that the first iterate V_1 can also be expressed in terms of $\tilde{V}_1$ and $\tilde{V}_2$ as well:

$$V_1 = \gamma_1 (A + \overline{\mu_1} I_n) \tilde{V}_1 = \gamma_1 \left(\overline{\mu_1} \tilde{V}_1 + \tilde{V}_2\right). \tag{20}$$

Using both iterates the LRCF is given by

$$Z = [V_1,\ V_2] = [\tilde{V}_1,\ \tilde{V}_2] \begin{bmatrix} \gamma_1 \overline{\mu_1} I_m & -\gamma_1 \overline{\mu_1} I_m \\ \gamma_1 I_m & \gamma_1 I_m \end{bmatrix}.$$

Now with $\tilde{Z} := [\tilde{V}_1,\ \tilde{V}_2]$ the approximate solution of (1) after these two initial steps is

$$ZZ^H = \tilde{Z} \begin{bmatrix} \gamma_1 \overline{\mu_1} I_m & -\gamma_1 \overline{\mu_1} I_m \\ \gamma_1 I_m & \gamma_1 I_m \end{bmatrix} \begin{bmatrix} \gamma_1 \mu_1 I_m & \gamma_1 I_m \\ -\gamma_1 \mu_1 I_m & \gamma_1 I_m \end{bmatrix} \tilde{Z}^T = \tilde{Z} \begin{bmatrix} 2\gamma_1^2 |\mu_1|^2 I_m & 0 \\ 0 & 2\gamma_1^2 I_m \end{bmatrix} \tilde{Z}^T,$$

such that Z can be constructed as

$$Z = \tilde{Z} \begin{bmatrix} \sqrt{2}\gamma_1 |\mu_1| I_m & 0 \\ 0 & \sqrt{2}\gamma_1 I_m \end{bmatrix} = \left[2\sqrt{-\mathrm{Re}\,(\mu_1)}|\mu_1| \tilde{V}_1,\ \ 2\sqrt{-\mathrm{Re}\,(\mu_1)} \tilde{V}_2\right].$$

Together with (19), (20) this results in Step 2 of [5, Algorithm 4.].

Next, we reformulate the equations defining the iteration in the case when after a pair of complex conjugate shifts parameters $\nu_{j-2} = \{\mu_{j-2},\ \mu_{j-1} := \overline{\mu_{j-2}}\}$ a real one μ_j occurs. Similar to the manipulations above, we concatenate the iterates belonging to ν_{j-2}:

$$V_{j-2} = \gamma_{j-2}(A + \mu_{j-2} I_n)^{-1}(A - \overline{\mu_{j-3}} I_n) V_{j-3},$$

where $\gamma_{j-2} := \sqrt{\frac{\operatorname{Re}(\mu_{j-2})}{\operatorname{Re}(\mu_{j-3})}}$, and thus

$$\begin{aligned} V_{j-1} &= (A + \overline{\mu_{j-2}} I_n)^{-1}(A - \overline{\mu_{j-2}} I_n) V_{j-2} \\ &= \gamma_{j-2}(A + \overline{\mu_{j-2}} I_n)^{-1}(A - \overline{\mu_{j-2}} I_n)(A + \mu_{j-2} I_n)^{-1}(A - \overline{\mu_{j-3}} I_n)\tilde{V}_{j-3} \\ &= \gamma_{j-2}(A - \overline{\mu_{j-2}} I_n)(A + \overline{\mu_{j-2}} I_n)^{-1}(A + \mu_{j-2} I_n)^{-1}(A - \overline{\mu_{j-3}} I_n)\tilde{V}_{j-3} \\ &= \gamma_{j-2}(A - \mu_{j-1} I_n)\left(A^2 + 2\operatorname{Re}(\mu_{j-1}) + |\mu_{j-1}|^2 I_n\right)^{-1}(A - \overline{\mu_{j-3}} I_n)\tilde{V}_{j-3} \\ &= \gamma_{j-2}(A - \mu_{j-1} I_n)\tilde{V}_{j-2} = \gamma_{j-2}\left(\tilde{V}_{j-2} - \mu_{j-1}\tilde{V}_{j-1}\right) \end{aligned} \tag{21}$$

with

$$\tilde{V}_{j-2} := \left(A^2 + 2\operatorname{Re}(\mu_{j-1}) + |\mu_{j-1}|^2 I_n\right)^{-1}(A - \overline{\mu_{j-3}} I_n)\tilde{V}_{j-3}, \quad \tilde{V}_{j-1} := A\tilde{V}_{j-2}.$$

Note that we assumed that $\tilde{V}_3$ is constructed using the appropriate equations in LRCF-ADI-R leading to real data only. This result is merged with the iterate associated to the current shift μ_j

$$V_j = \gamma_j (A + \mu_j I_n)^{-1}(A - \overline{\mu_{j-1}} I_n) V_{j-1},$$

where $\gamma_j := \sqrt{\frac{\mu_j}{\operatorname{Re}(\mu_{j-1})}}$, so that

$$\begin{aligned} V_j &= \gamma_j\gamma_{j-2}(A + \mu_j I_n)^{-1}(A - \overline{\mu_{j-1}} I_n)(A - \mu_{j-1} I_n)\tilde{V}_{j-2} \\ &= \gamma_j\gamma_{j-2}\left(I - (\mu_j + \overline{\mu_{j-1}})(A + \mu_j I_n)^{-1})\right)(A - \mu_{j-1} I_n)\tilde{V}_{j-2} \\ &= \gamma_j\gamma_{j-2}\left((A - \mu_{j-1} I_n)\tilde{V}_{j-2} - (\mu_j + \overline{\mu_{j-1}})(A + \mu_j I_n)^{-1}(A - \mu_{j-1} I_n)\right)\tilde{V}_{j-2} \\ &= \gamma_j\gamma_{j-2}\left((A - \mu_{j-1} I_n)\tilde{V}_{j-2} - (\mu_j + \overline{\mu_{j-1}})(I - (\mu_j + \mu_{j-1})(A + \mu_j I_n)^{-1}\right)\tilde{V}_{j-2} \\ &= \gamma_j\gamma_{j-2}\left(\tilde{V}_{j-1} - \mu_{j-1}\tilde{V}_{j-2} - (\mu_j + \overline{\mu_{j-1}})\tilde{V}_{j-2} - (\mu_j + \overline{\mu_{j-1}})(\mu_j + \mu_{j-1})(A + \mu_j I_n)^{-1}\right)\tilde{V}_{j-2} \\ &= \gamma_j\gamma_{j-2}\left(\tilde{V}_{j-1} - (2\operatorname{Re}(\mu_{j-1}) + \mu_j)\tilde{V}_{j-2} - (|\mu_{j-1}|^2 + 2\mu_j\operatorname{Re}(\mu_{j-1}) + \mu_j^2)(A + \mu_j I_n)^{-1}\right)\tilde{V}_{j-2} \;{}^{6} \\ &= \gamma_j\gamma_{j-2}\tilde{V}_j \end{aligned} \tag{22}$$

with

$$\tilde{V}_j := \left(\tilde{V}_{j-1} - (2\operatorname{Re}(\mu_{j-1}) + \mu_j)\tilde{V}_{j-2} - (|\mu_{j-1}|^2 + 2\mu_j\operatorname{Re}(\mu_{j-1}) + \mu_j^2)(A + \mu_j I_n)^{-1}\right)\tilde{V}_{j-2}.$$

Note that the denominator in

$$\gamma_j\gamma_{j-2} = \sqrt{\frac{\mu_j\operatorname{Re}(\mu_{j-2})}{\operatorname{Re}(\mu_{j-1})\operatorname{Re}(\mu_{j-3})}} = \sqrt{\frac{-2\mu_j}{-2\operatorname{Re}(\mu_{j-3})}}$$

will cancel out due to the scalar factor in front of the iterate V_{j-3}. Hence, the new LRCF becomes $Z_j = [Z_{j-1},\ \sqrt{-2\mu_j}\tilde{V}_j]$ and we have eventually obtained the equations of Step 4 in [5, Algorithm 4.].

Now we address the opposite case when after a real shift μ_{j-1} a pair of complex conjugated ones $\nu_j := \{\mu_j,\ \mu_{j+1} := \mu_j\}$ follows.

Merging again both iterates $V_j,\ V_{j+1}$ with respect to ν_j leads with $\gamma_j := \sqrt{\frac{\operatorname{Re}(\mu_j)}{\mu_{j-1}}}$ to

$$\begin{aligned} V_j &= \gamma_j(A + \mu_j I_n)^{-1}(A - \mu_{j-1})\tilde{V}_{j-1} \\ V_{j+1} &= (A + \overline{\mu_j} I_n)^{-1}(A - \overline{\mu_j} I_n) V_j \\ &= \gamma_j(A + \overline{\mu_j} I_n)^{-1}(A - \overline{\mu_j} I_n)(A + \mu_j I_n)^{-1}(A - \mu_{j-1})\tilde{V}_{j-1} \\ &= \gamma_j(A - \overline{\mu_j} I_n)\left(A^2 + 2\operatorname{Re}(\mu_j) + |\mu_j|^2 I_n\right)^{-1}(A - \mu_{j-1})\tilde{V}_{j-1}, \\ &= -\overline{\mu_j}\gamma_j\tilde{V}_j + \gamma_j\tilde{V}_{j+1} \end{aligned} \tag{23}$$

$$\text{with}\quad \tilde{V}_j := \left(A^2 + 2\operatorname{Re}(\mu_j) + |\mu_j|^2 I_n\right)^{-1}(A - \mu_{j-1})\tilde{V}_{j-1}, \quad \tilde{V}_{j+1} := A\tilde{V}_j,$$

[6]We note that the 2 in front of $\operatorname{Re}(\mu_{j-1})$ appears to be in the wrong position in Step 4 of [5, Algorithm 4.]

and $\tilde{V}_{j-1}$ being the real result of Steps 3. or 4. of LRCF-ADI-R. As before, V_j can be expressed as

$$V_j = \gamma_j(A + \overline{\mu_j} I_n)\tilde{V}_j = \overline{\mu_j}\gamma_j\tilde{V}_j + \gamma_j\tilde{V}_{j+1}. \tag{24}$$

These are the same relations as (19), (20) in the situation investigated first and thus, the updated LRCF is

$$Z_{j+1} = \left[Z_{j-1},\ 2\sqrt{-\operatorname{Re}(\mu_j)}|\mu_j|\tilde{V}_j,\ 2\sqrt{-\operatorname{Re}(\mu_j)}\tilde{V}_{j+1}\right]$$

which directly reveals Step 6 of [5, Algorithm 4.].

Finally, we look into the situation of two subsequent pairs of complex conjugated iterates. Let the first pair be $\mu_{j-2},\ \mu_{j-1} := \overline{\mu_{j-2}}$ and $\mu_j,\ \mu_{j+1} := \overline{\mu_j}$ the second one. Following the same steps as in the previous discussed case, the iterates for the first pair satisfy

$$V_{j-1} = \gamma_{j-2}(A - \mu_{j-1}I_n)\tilde{V}_{j-2} = -\gamma_{j-2}\mu_{j-1}\tilde{V}_{j-1} + \gamma_{j-2}\tilde{V}_{j-2},$$

where $\tilde{V}_{j-1},\ \tilde{V}_{j-2}$ are defined by (21) and $\gamma_{j-2} := \sqrt{\frac{\operatorname{Re}(\mu_{j-2})}{\operatorname{Re}(\mu_{j-3})}}$. The iterates using the second pair obey with $\gamma_j := \sqrt{\frac{\operatorname{Re}(\mu_j)}{\operatorname{Re}(\mu_{j-1})}}$

$$\begin{aligned}
V_j &= \gamma_j(A + \mu_j I_n)^{-1}(A - \overline{\mu_{j-1}}I_n)V_{j-1},\\
V_{j+1} &= (A + \overline{\mu_j}I_n)^{-1}(A - \overline{\mu_j}I_n)V_j\\
&= \gamma_j(A + \overline{\mu_j}I_n)^{-1}(A - \overline{\mu_j}I_n)(A + \mu_j I_n)^{-1}(A - \overline{\mu_{j-1}}I_n)V_{j-1}\\
&= \gamma_j(A - \overline{\mu_j}I_n)(A^2 + 2\operatorname{Re}(\mu_j)A + |\mu_j|^2 I_n)^{-1}(A - \overline{\mu_{j-1}}I_n)V_{j-1}\\
&= \gamma_j\gamma_{j-2}(A - \overline{\mu_j}I_n)(A^2 + 2\operatorname{Re}(\mu_j)A + |\mu_j|^2 I_n)^{-1}(A - \overline{\mu_{j-1}}I_n)(A - \mu_{j-1}I_n)\tilde{V}_{j-2}\\
&= -\gamma_j\gamma_{j-2}\overline{\mu_j}\tilde{V}_j + \gamma_j\gamma_{j-2}\tilde{V}_{j+1}
\end{aligned} \tag{25}$$

using

$$\tilde{V}_j := (A^2 + 2\operatorname{Re}(\mu_j)A + |\mu_j|^2 I_n)^{-1}(A - \overline{\mu_{j-1}}I_n)(A - \mu_{j-1}I_n)\tilde{V}_{j-2}, \quad \tilde{V}_{j+1} := A\tilde{V}_j.$$

Now we invoke to following reformulations

$$\begin{aligned}
(A - \overline{\mu_{j-1}}I_n)(A - \mu_{j-1}I_n) &= (A + (\overline{\mu_j} - \overline{\mu_j} - \overline{\mu_{j-1}})I_n)\,(A + (\mu_j - \mu_j - \mu_{j-1})I_n)\\
&= (A - \overline{\mu_j}I_n)(A - \mu_j I_n) - [\mu_j + \mu_{j-1} + \overline{\mu_j} + \overline{\mu_{j-1}}]\,A\\
&\quad - \mu_j(\overline{\mu_j} + \overline{\mu_{j-1}})I_n - \overline{\mu_j}(\mu_j + \mu_{j-1})I_n + (\overline{\mu_j} + \overline{\mu_{j-1}})(\mu_j + \mu_{j-1})I_n\\
&= (A^2 + 2\operatorname{Re}(\mu_j)A + |\mu_j|^2 I_n) - 2\operatorname{Re}(\mu_j + \mu_{j-1})A + (|\mu_{j-1}|^2 - |\mu_j|^2)I_n
\end{aligned}$$

which lead to

$$\tilde{V}_j = \tilde{V}_{j-2} + (A^2 + 2\operatorname{Re}(\mu_j)A + |\mu_j|^2 I_n)^{-1}\left[(|\mu_{j-1}|^2 - |\mu_j|^2)\tilde{V}_{j-2} - 2\operatorname{Re}(\mu_j + \mu_{j-1})\tilde{V}_{j-1}\right]. \tag{26}$$

Similar as in the cases before, we have relation (25) and

$$V_j = \gamma_j\gamma_{j-2}\overline{\mu_j}\tilde{V}_j + \gamma_j\gamma_{j-2}\tilde{V}_{j+1}. \tag{27}$$

Using the same argumentation concerning the scalar factors $\gamma_j,\ \gamma_{j-2}$ as above, we are able to find the correct augmentation of the LRCFs

$$Z_{j+1} = \left[Z_{j-1},\ 2\sqrt{-\operatorname{Re}(\mu_j)}|\mu_j|\tilde{V}_j,\ 2\sqrt{-\operatorname{Re}(\mu_j)}\tilde{V}_{j+1}\right]. \tag{28}$$

Together with (26), (25), (27) this results exactly in Steps 7 and 8 of [5, Algorithm 4.].

References

[1] H.T. Banks and K. Ito. A numerical algorithm for optimal feedback gains in high dimensional linear quadratic regulator problems. *SIAM J. Cont. Optim.*, 29(3):499–515, 1991. 11

[2] R.H. Bartels and G.W. Stewart. Solution of the matrix equation $AX + XB = C$: Algorithm 432. *Comm. ACM*, 15:820–826, 1972. 1, 4

[3] P. Benner. Solving large-scale control problems. *IEEE Control Systems Magazine*, 14(1):44–59, 2004. 8, 9

[4] P. Benner and H. Faßbender. On the numerical solution of large-scale sparse discrete-time Riccati equations. *Adv. Comput. Math.*, 35:119–147, 2011. 10.1007/s10444-011-9174-7. 16

[5] P. Benner, J.-R. Li, and T. Penzl. Numerical solution of large Lyapunov equations, Riccati equations, and linear-quadratic control problems. *Numer. Lin. Alg. Appl.*, 15(9):755–777, 2008. 4, 11, 12, 14, 17, 18, 19

[6] P. Benner, R.C. Li, and N. Truhar. On the ADI Method for Sylvester Equations. *J. Comput. Appl. Math.*, 233:1035–1045, December 2009. 16

[7] P. Benner and H. Mena. BDF methods for large-scale differential Riccati equations. In B. De Moor, B. Motmans, J. Willems, P. Van Dooren, and V. Blondel, editors, *Proc. 16th Intl. Symp. Mathematical Theory of Network and Systems, MTNS 2004*, 2004. 12 p., available at http://www.mtns2004.be. 12

[8] P. Benner and H. Mena. Rosenbrock methods for solving differential Riccati equations. Technical Report MPIMD/11-06, Max Planck Institute Magdeburg Preprints, October 2011. 12

[9] P. Benner and E.S. Quintana-Ortí. Solving stable generalized Lyapunov equations with the matrix sign function. *Numer. Algorithms*, 20(1):75–100, 1999. 4

[10] P. Benner and J. Saak. A Galerkin-Newton-ADI Method for Solving Large-Scale Algebraic Riccati Equations. Preprint SPP1253-090, SPP1253, January 2010. http://www.am.uni-erlangen.de/home/spp1253/wiki/index.php/Preprints. 3, 4, 11

[11] P. Benner and J. Saak. Efficient Balancing based MOR for Large Scale Second Order Systems. *Math. Comput. Model. Dyn. Sys.*, 17(2):123–143, 2011. DOI:10.1080/13873954.2010.540822. 9

[12] C. H. Bischof and G. Quintana-Ortí. Algorithm 782: Codes for rank-revealing QR factorizations of dense matrices. *ACM Trans. Math. Software*, 24(2):254–257, 1998. 4

[13] T. A. Davis. *Direct Methods for Sparse Linear Systems (Fundamentals of Algorithms 2)*. SIAM, Philadelphia, PA, USA, 2006. 2

[14] I.S. Duff, A.M. Erisman, and J.K. Reid. *Direct methods for sparse matrices*. Clarendon Press, Oxford, UK, 1989. 2

[15] F. Freitas, N. Martins, S. L. Varricchio, J. Rommes, and F. C. Veliz. Reduced-order transfer matrices from RLC network descriptor models of electric power grids. *IEEE Trans. Power Systems*, PP(99), 2011. 10

[16] F. Freitas, J. Rommes, and N. Martins. Gramian-based reduction method applied to large sparse power system descriptor models. *IEEE Trans. Power Systems*, 23(3):1258–1270, August 2008. 10, 13, 15

[17] Z. Gajić and M. Qureshi. *Lyapunov Matrix Equation in System Stability and Control*. Math. in Science and Engineering. Academic Press, San Diego, CA, 1995. 1

[18] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, third edition, 1996. 11

[19] L. Grasedyck. Existence of a low rank or H-matrix approximant to the solution of a Sylvester equation. *Numer. Lin. Alg. Appl.*, 11:371–389, 2004. 1

[20] S.J. Hammarling. Numerical solution of the stable, non-negative definite Lyapunov equation. *IMA J. Numer. Anal.*, 2:303–323, 1982. 1, 4

[21] D.L. Kleinman. On an iterative technique for Riccati equation computations. *IEEE Trans. Automat. Control*, AC-13:114–115, 1968. 1, 11

[22] J.-R. Li and J. White. Low rank solution of Lyapunov equations. *SIAM J. Matrix Anal. Appl.*, 24(1):260–280, 2002. 2

[23] N. Martins, L.T.G. Lima, and H.J.C.P. Pinto. Computing dominant poles of power system transfer functions. *IEEE Trans. Power Systems*, 11(1):162–170, 1996. 2

[24] V. Mehrmann and T. Stykel. Balanced truncation model reduction for large-scale systems in descriptor form. In P. Benner, V. Mehrmann, and D.C. Sorensen, editors, *Dimension Reduction of Large-Scale Systems*, volume 45 of *Lecture Notes in Computational Science and Engineering*, pages 83–115. Springer-Verlag, Berlin/Heidelberg, Germany, 2005. 10

[25] B. C. Moore. Principal component analysis in linear systems: controllability, observability, and model reduction. *IEEE Trans. Automat. Control*, AC-26(1):17–32, 1981. 1, 3

[26] T. Penzl. A cyclic low rank Smith method for large sparse Lyapunov equations. *SIAM J. Sci. Comput.*, 21(4):1401–1418, 2000. 2, 4, 13

[27] T. Penzl. LYAPACK Users Guide. Technical Report SFB393/00-33, Sonderforschungsbereich 393 *Numerische Simulation auf massiv parallelen Rechnern*, TU Chemnitz, 09107 Chemnitz, Germany, 2000. Available from http://www.tu-chemnitz.de/sfb393/sfb00pr.html. 4, 5, 11

[28] J.D. Roberts. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Internat. J. Control*, 32:677–687, 1980. (Reprint of Technical Report No. TR-13, CUED/B-Control, Cambridge University, Engineering Department, 1971). 1

[29] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, PA, USA, 2003. 2

[30] J. Saak. *Efficient Numerical Solution of Large Scale Algebraic Matrix Equations in PDE Control and Model Order Reduction*. PhD thesis, TU Chemnitz, July 2009. Available from http://nbn-resolving.de/urn:nbn:de:bsz:ch1-200901642. 2, 3, 4, 8, 9, 11

[31] V. Simoncini. A new iterative method for solving large-scale Lyapunov matrix equations. *SIAM J. Sci. Comput.*, 29(3):1268–1288, 2007. 3, 13

[32] D.C. Sorensen and Y. Zhou. Bounds on eigenvalue decay rates and sensitivity of solutions to Lyapunov equations. Technical Report TR02-07, Dept. of Comp. Appl. Math., Rice University, Houston, TX, June 2002. Available online from http://www.caam.rice.edu/caam/trs/tr02.html#TR02. 1

[33] D.C. Sorensen and Y. Zhou. Direct methods for matrix Sylvester and Lyapunov equations. *J. Appl. Math*, 2003:277–303, 2002. 1, 4

[34] F. Tisseur and K. Meerbergen. The quadratic eigenvalue problem. *SIAM Review*, 43(2):235–286, 2001. 9

[35] N. Truhar and K. Veselic. An efficient method for estimating the optimal dampers' viscosity for linear vibrating systems using Lyapunov equation. *SIAM J. Matrix Anal. Appl.*, 31(1):18–39, 2009. 13

[36] H. A. Van der Vorst. *Iterative Krylov Methods for Large Linear Systems*. Cambridge University Press, Cambridge, 2003. 2

[37] E.L. Wachspress. Iterative solution of the Lyapunov matrix equation. *Appl. Math. Letters*, 107:87–90, 1988. 2

[38] E.L. Wachspress. The ADI model problem, 1995. Available from the author. 2